TITLE: METHOD AND APPARATUS FOR VIDEO UNDERFLOW DETECTION IN A RASTER ENGINE

REFERENCE TO RELATED APPLICATION

This application is a continuation-in-part of co-pending U.S. Patent Application Serial No. 09/672,632, which was filed September 28, 2000, entitled RASTER ENGINE WITH BOUNDED VIDEO SIGNATURE ANALYZER.

TECHNICAL FIELD

The present invention relates generally to the field of video displays and more particularly to improved methods and apparatus for video underflow detection in a raster engine.

BACKGROUND OF THE INVENTION

Video displays are used in computer systems to present visual images to a user based on video data provided by a computer or other processing device. The display allows a user to effectively receive information from and to interact with application programs running in the system. Such computer systems and displays are employed in numerous business, consumer, entertainment, and industrial settings, including automated industrial control systems.

Displays are available in a variety of forms, such as color or monochrome, flat panel, liquid crystal display (LCD), electro-luminescent (EL), plasma display panels (PDP), vacuum fluorescent displays (VFD), cathode ray tube (CRT), and may be interfaced to a computer system in analog or digital fashion. The display is provided with video data frame by frame, which is scanned onto the display screen according to a scanning method which may include progressive scan, dual scan, interleave scan, or interlaced scanning. The cost of displays varies with the display resolution and quality. For example, color displays generally cost more than monochrome displays. The number of pixels, as well as the number of available colors per pixel (bits per pixels) also affects display cost. The cost of a computer display may be a large percentage of the overall computer system cost. As the application of computer system displays varies greatly, displays are accordingly provided in a variety of price ranges.

Interfacing between a computer or other processing device and a display is

ordinarily accomplished using a video controller, also variously referred to as graphics adapter, graphics controller, video display adapter, display controller, and display adapter. The screen resolution on a PC is determined by the video controller, which may be plugged into one of the computer's expansion slots. In conventional systems, the display must also be able to adjust to the resolution of the video controller. Common video controllers come with their own drivers for an operating system, which are installed after the video controller is installed. The driver allows the operating system to display its video output at a certain number of resolutions and colors. The video controller may include a raster engine which rasterizes video data from a frame buffer into a format that the display can accept for rendering to a user.

Raster engines typically obtain image data from a frame buffer in memory via a bus, wherein the frame buffer may be in main memory or in a separate display memory. The bus may provide access between the raster engine and the main memory, as well as between other devices in a computer system. In this shared system bus configuration, situations may arise in which the raster engine requires display image data from the frame buffer, and yet the raster engine cannot timely obtain such data due to contention with other devices using the common or shared bus. As a result, the raster engine may become empty, for example, during excessive bus loading conditions. The video display interfaced by the raster engine may exhibit undesirable visual effects under these conditions. For example, the display may suffer from visual defects such as jittering, shifting, flashing, and blank-outs in the displayed video image. Such undesirable visual defects are also experienced when a raster engine on an isolated or dedicated bus becomes starved for video data. Thus, there is a need for improved methods and apparatus for preventing or minimizing empty raster engine conditions, and the undesirable display effects associated therewith.

SUMMARY OF THE INVENTION

The following presents a simplified summary of the invention in order to provide a basic understanding of some aspects of the invention. This summary is not an extensive overview of the invention. It is intended to neither identify key or critical elements of the invention nor delineate the scope of the invention. Its sole purpose is to present some concepts of the invention in a simplified form as a prelude to the more detailed description that is presented later.

The foregoing and other shortcomings associated with conventional video controller devices and methodologies are reduced or minimized by the present invention, which provides a video controller and raster engine which is easily programmed to interface a computer system running a variety of application programs with a plurality of disparate display types. The invention may thus be employed in high end as well as highly cost sensitive computer system applications in association with displays ranging from high definition television (HDTV) to low resolution monochrome EL and/or LCD display panels.

The invention provides for software programmable registers in the video controller raster engine by which a user may programmatically adapt or configure the raster engine to provide video data to a wide variety of different displays with different color capabilities and resolutions. The raster engine comprises an underflow detection system, which may provide an indication of current or anticipated underflow conditions, which may be provided to a system processor or other device for taking some steps toward remedying the cause of the underflow. In addition, programmable grayscaling is provided, together with hardware cursor features applicable to dual scan displays, and hardware blinking apparatus providing low overhead blinking on an individual pixel basis. Moreover, the invention provides for integrating a video signature analyzer in the video controller, providing for self-testing, as well as the capability of testing video signatures for displays having changing portions.

According to an aspect of the present invention, the raster engine may provide an indication to a host processor that the raster engine is underflowing or about to underflow, or that a lockup condition exists in the raster engine. Input and output counters in the raster engine first in first out (FIFO) memory system, which interfaces the host bus with the raster engine video systems, are read by an underflow detection system which is adapted to provide an underflow indication according to the counter values. The underflow detection and indication system thus minimizes or reduces the undesirable visual effects associated with a starved or empty raster engine, and allows remedial and/or notification measures to be taken in a computer system employing the raster engine.

The video controller raster engine receives video data from a frame buffer and renders formatted data to a display in a computer system. According to another aspect of the invention, the raster engine comprises a first in first out (FIFO) memory interfacing a

host bus in the computer system with the raster engine. The FIFO memory obtains video data from the frame buffer via the host bus and provides video data to a video pipeline in the raster engine. Input and output counters are provided to indicate the data in the FIFO as data is received from the frame buffer and fed to the video pipeline. The input counter has a value indicating video data obtained from the frame buffer, and the output counter has a value indicating video data provided to the video pipeline. The counters may operate from separate clocks. For example, the input counter may operate according to a host clock, and the output counter may operate according to a video clock. The raster engine further comprises a control logic system associated with the FIFO memory to selectively provide an underflow indication according to the input and output counter values.

The underflow indication may be used by a system processor to provide extra bus bandwidth or take some other action to reduce or prevent the FIFO memory from becoming starved for video data, thereby reducing the occurrence of deleterious visual display defects associated with raster engine starvation. The control logic system may comprise various hardware and/or software in order to compare the input and output counter values to determine if an underflow condition exists or is about to exist in the raster engine FIFO memory. For example, the logic system may determine the difference between the input and output counters, and compare the counter difference with a threshold value, which may be obtained from a programmable register. Thus, where the counters are of equal value, the logic system may determine and provide an indication (*e.g.*, a video underflow interrupt signal) that an underflow condition exists. In addition, where a small difference (*e.g.*, less than or equal to a threshold value) exists between the input and output counters, the logic system may determine that an underflow situation is about to occur, and provide a corresponding indication. In order to ensure valid detection of underflow conditions where two separate clocks are used in association with the FIFO memory, the underflow indication may be provided if the difference value is less than or equal to the threshold for a number of consecutive cycles of a host clock (*e.g.*, two clock cycles).

The invention further comprises a methodology for detecting and/or indicating an underflow condition (*e.g.*, or the likelihood of such a condition occurring) in a raster engine. The method comprises obtaining an input counter value indicative of video data obtained from a frame buffer, and obtaining an output counter value indicative of video

data provided from a memory to a video pipeline in the raster engine. The method further comprises performing a comparison of the input and output counter values, and selectively providing an underflow indication according to the input and output counter value comparison. The comparison of the counter values may include subtracting the input counter value from the output counter value to obtain a difference value, and comparing the difference value with a threshold value. An underflow signal may accordingly be provided if the input and output counter values are within a threshold value of each other (*e.g.*, if the difference value is less than or equal to the threshold). In order to ensure valid detection of underflow conditions where two separate clocks are used in association with the FIFO memory, the underflow indication may be provided if the difference value is less than or equal to the threshold for a number of consecutive cycles of a host clock (*e.g.*, two clock cycles).

To the accomplishment of the foregoing and related ends, certain illustrative aspects of the present invention are hereinafter described with reference to the attached drawing figures. The following description and the annexed drawings set forth in detail certain illustrative applications and aspects of the invention. These are indicative, however, of but a few of the various ways in which the principles of the invention may be employed. Other aspects, advantages and novel features of the invention will become apparent from the following detailed description of the invention when considered in conjunction with the drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

The foregoing and other aspects of the invention will become apparent from the following detailed description of various aspects of the invention and the attached drawings in which:

Fig. 1 is a schematic diagram illustrating an exemplary raster engine in accordance with the present invention;

Fig. 2A is a schematic diagram illustrating a computer system in which various aspects of the invention may be employed;

Fig. 2B is a schematic diagram further illustrating the raster engine of Fig. 1;

Fig. 3 is a schematic diagram illustrating an exemplary signature analyzer in accordance with an aspect of the invention;

Fig. 4 is a schematic diagram illustrating an exemplary linear feedback shift

register in accordance with another aspect of the invention;

Fig. 5 is a schematic diagram illustrating a bounded video signature analysis for a bounded portion of a display using the exemplary signature analyzer of Fig. 3;

Figs. 6A-6E are schematic diagrams illustrating exemplary control and/or data registers associated with the exemplary signature analyzer of Fig. 3;

Fig. 7A is a schematic diagram illustrating an exemplary cursor image in accordance with another aspect of the invention;

Fig. 7B is a schematic diagram illustrating an exemplary progressive scan display including the cursor image of Fig. 7A;

Fig. 8A is a schematic diagram illustrating another exemplary cursor image in accordance with the invention;

Fig. 8B is a schematic diagram illustrating an exemplary dual scan display including the cursor image of Fig. 8A;

Fig. 9A is a schematic diagram illustrating another exemplary cursor image in accordance with the invention;

Fig. 9B is a schematic diagram illustrating the exemplary dual scan display of Fig. 8B including the cursor image of Fig. 9A;

Fig. 10 is a flow diagram illustrating an exemplary method in accordance with another aspect of the invention;

Figs. 11A-11G are schematic diagrams illustrating exemplary control and/or data registers associated with the hardware cursor controller of Fig. 1;

Fig. 12 is a schematic diagram illustrating an exemplary color mux and associated control registers in accordance with another aspect of the invention;

Figs. 13A-13C are schematic diagrams illustrating exemplary control and/or data registers associated with the color mux of Fig. 12;

Figs. 14A and 14B illustrated an exemplary pixel transfer mapping in accordance with another aspect of the invention;

Fig. 15 is a schematic diagram illustrating an exemplary hardware blinking apparatus in accordance with another aspect of the invention;

Figs. 16A-16E are schematic diagrams illustrating exemplary control and/or data registers associated with the hardware blinking apparatus of Fig. 15;

Fig. 17 is a schematic diagram illustrating an exemplary grayscale generator in accordance with another aspect of the invention;

Fig. 18 is a schematic diagram illustrating several exemplary counters associated with the grayscale generator of Fig. 17;

Fig. 19 is a schematic diagram illustrating an exemplary control register associated with the grayscale generator of Figs. 17 and 18;

Fig. 20 is a schematic diagram illustrating an exemplary programmable grayscale look up table matrix in accordance with another aspect of the invention;

Fig. 21 is a schematic diagram illustrating another exemplary programmable grayscale look up table matrix in accordance with the invention;

Fig. 22 is a schematic diagram illustrating an exemplary 4x4x4 grayscale pattern in accordance with the invention;

Fig. 23 is a schematic diagram illustrating another exemplary 4x4x4 grayscale pattern in accordance with the invention;

Fig. 24 is a schematic diagram illustrating another exemplary 4x4x4 grayscale pattern in accordance with the invention;

Fig. 25 is a schematic diagram illustrating another exemplary programmable grayscale look up table matrix in accordance with the invention;

Fig. 26 is a schematic diagram illustrating an exemplary 3x3x3 grayscale pattern in accordance with the invention;

Fig. 27 is a schematic diagram illustrating another exemplary 3x3x3 grayscale pattern in accordance with the invention;

Fig. 28 is a schematic diagram illustrating another exemplary programmable grayscale look up table matrix in accordance with the invention;

Fig. 29 is a schematic diagram illustrating an exemplary 4x3x3 grayscale pattern in accordance with the invention;

Fig. 30 is a schematic diagram illustrating another exemplary programmable grayscale look up table matrix in accordance with the invention;

Fig. 31 is a table illustrating several exemplary raster engine output modes in accordance with the invention;

Fig. 32 is a schematic diagram illustrating an exemplary computer system in which various aspects of the present invention may be carried out;

Fig. 33 is schematic diagram illustrating another computer system including an exemplary raster engine providing an underflow indication in accordance with another aspect of the invention;

Fig. 34 is a schematic diagram illustrating further details of the exemplary raster engine of Fig. 33;

Fig. 35 is a schematic diagram illustrating an exemplary underflow detection system in accordance with an aspect of the invention;

Fig. 36 is a schematic diagram illustrating another exemplary underflow detection system in accordance with the invention; and

Fig. 37 is a flow diagram illustrating an exemplary method of detecting underflow conditions in accordance with another aspect of the invention.

DETAILED DESCRIPTION OF THE INVENTION

The following is a detailed description of the present invention made in conjunction with the attached figures, wherein like reference numerals will refer to like elements throughout. According to the invention, an improved raster engine is provided to render video data from a frame buffer to one of a plurality of disparate displays which comprises an integral bounded video signature analyzer, a hardware cursor apparatus supporting dual scanned displays, programmatic support for multiple disparate display types, multi-mode programmable hardware blinking, programmable multiple color depth digital display interface, and programmable matrix controlled grayscale generation.

Referring now to the drawings, Fig. 1 illustrates an exemplary raster engine 2, which is adapted to provide data and interface signals for a variety of displays, including analog CRTs and digital LCDs (not shown). In addition, the raster engine 2 has fully programmable video interface timing for progressive non-interlaced, dual scanning, line interleaved, and interlaced displays. Programmable compare and register logic 4 allows a user or a host system application program to select appropriate display modes for interfacing a frame buffer with one or a plurality of disparate display devices. Compare and register logic 4 may comprise one or more of the control registers illustrated and described hereinafter. Separate DAC interface signals are provided to allow analog red, green, blue (RGB) signal generation for analog LCD displays or CRTs. Raster engine 2 is also designed to generate CCIR656 4:2:2 YCrCb digital video output signals for optionally interfacing an NTSC encoder (not shown). Raster engine 2 further advantageously provides support for an 8-bit parallel display interface for interfacing to low-end display modules with integrated controller and frame buffer, and may also comprise an integrated triple 8-bit DAC 6 for directly supporting analog output to CRT

displays.

As illustrated in Fig. 1, the raster engine 2 includes a video pipeline comprising several major sections; a video image line output scanner and transfer interface (VILOSATI) 14, a video first in first out system (FIFO) 16, a pixel mux 18, a blink logic system 8, a dual color look up table (LUT) 10, a grayscale generator 12, an RGB color mux 20, a pixel shift logic system 22, hardware cursor logic system 24, a YCrCb encoder 26, a video timing section comprising horizontal and vertical counters 28, and the compare and register logic 4. In addition, a video stream signature analyzer 30 may be integrated in the raster engine 2 for built in self testing. The FIFO 16 further comprises a dual port RAM device 32, input address counters 34, an output address counter 36, and control logic 38 for interfacing with the VILOSATI 14. The FIFO control logic 38 further comprises an underflow interrupt output adapted to indicate a current or potential underflow condition in the FIFO 16. An output mux 40 selectively provides output video data from one of the YCrCb encoder 26 and the pixel shift logic system 22 via data and clock buffers 42 and 44, respectively. The hardware cursor system 24 comprises an AMBA cursor bus master 50 for controlling the transfer of cursor data, cursor address counters 52, cursor state machines 54, cursor output counters 56, and a cursor line buffer 58.

Referring also to Fig. 2A, an exemplary computer system 60 is illustrated having a central processing unit (CPU) 62, a memory 64, and a bus 66 providing an interface therebetween. A video frame buffer 68 may interface with the bus 66 via a bus interface 70, or may alternatively be provided in a portion of main memory 64, wherein the beginning of video lines may be located on any 32 bit word boundary. Raster engine 2 may be operatively connected with the bus 66 for receiving video data therefrom for rendering to a display device 72. In addition, the bus 66 (*e.g.*, including address and data busses) may provide access to the various control registers in raster engine 2, including compare and register logic 4. Video screen start registers (not shown) may be used to determine the upper left corner of the video screen. Video word addressing in screen memory may be from left to right and then top to bottom.

Four bit pixels packaged within video words may be organized in device independent bitmap (DIB) format with the left most pixel in the most significant location on a per byte basis. Several screens may be available for video display depending on screen size, pixel depth, and amount of memory dedicated to video images. The screen

size may be up to 4096 x 4096 pixels and the pixel depth may be 4, 8, 16, 24, or 32 bpp. The raster engine 2 provides a pulse width modulated brightness control output that can be used in conjunction with a resistor and capacitor (not shown) to provide a DC voltage level for brightness control. The signal may be further employed for direct pulse width modulated cold cathode fluorescent lamp (CCFL) brightness control that can be synchronized to a display frame rate.

The raster engine 2 pipeline includes a hardware pixel blink logic system 8, adapted to selectively blink pixels on a display according to a programmable count of vertical sync intervals in a BLINKRATE register, as described in greater detail hereinafter. For 4 bpp and 8 bpp modes, either multiple or single bit planes may be used to specify blinking pixels according to the 256x24 SRAM look up table 10. This allows the number of definable blinking pixels to range from all pixel combinations blinking to one pixel combination blinking, providing significant overhead savings over conventional software blinking techniques, and finer grained blinking control than was available using conventional character blinking methodologies. For 16 bpp and 24 bpp modes, the blink logic system 8 may bypass the look up table 10, whereby blink functions may be accomplished via logic transformations of pixel data. In addition to logical AND/OR/XOR LUT address translations, the system 8 will support logical blink to background, blink dimmer, blink brighter, and blink to reverse operation.

The raster engine 2 may further comprise a dual look up table (LUT) 10, wherein each LUT will allow the raster engine 2 to output 256 different pixel combinations of 24 bit pixels in lower color depth modes. The raster engine 2 is further adapted to support video information as DIB format stored in a packed pixel architecture, although the video information need not be stored in a packed line architecture. The raster engine 2 allows a different memory organization between video scan out and graphic image memory. Therefore, memory gaps may exist between lines. Accordingly, the graphics memory may be organized wider than the video frame. For example, this may be used for left and right panning of the displayed information.

The grayscale generator 12 is adapted to generate grayscales on monochrome (or color) display types. The grayscale generator 12 supports up to 8 grayscale shades including on and off, by dithering pixels based on frame count, screen location, and pixel value. For example, the pixel value may be determined by the least significant 3 bits from LUT translated pixel data for any bpp mode. The raster engine 2 loads image data

from a special DMA interface to a DRAM memory controller, and further comprises a separate advanced high speed bus (AHB) bus master for collecting hardware cursor information from anywhere in a host computer system memory.

The raster engine 2 also provides hardware cursor support via hardware cursor logic system 24. System 24 comprises an AMBA cursor bus master 50, cursor address counters 52, cursor state machines 54, cursor output counters 56, and a cursor line buffer 58. The cursor image size is adjustable to 16, 32, 48, or 64 pixels wide by up to 64 pixels in height, and is stored anywhere in memory as a 2 bpp format. The image pixel information implies transparent, inverted, cursor color 1, or cursor color 2. The cursor hardware may be supplied an image starting address, 2 cursor colors, an X and Y screen location, and a cursor size. Using this information, the raster engine 2 overlays the cursor in the output video stream. Bottom and right edge clipping may also be performed by the raster engine hardware. The raster engine 2 further provides hardware cursor support for dual scan display types according to a selected display mode, as described in greater detail hereinafter.

The VILOSATI 14 connects to a dedicated DMA port on an SDRAM controller (not shown) and reads video image data from memory, such as a frame buffer, and thereafter transfers the image data to the video FIFO 16. VILOSATI 14 keeps track of image location, width, and depth for both progressive and dual scanned images, and responds to controls (*e.g.*, FULL, DS_FULL) from the FIFO 16 for more video data. During single scan operation, when the FIFO 16 has room for a 16 word burst, the FULL signal is inactive and VILOSATI 14 attempts to initiate a burst. The VILOSATI 14 will initiate appropriate size transfers and bursts in order to get to a 16 word boundary. After this point, VILOSATI 14 will perform transfers more efficiently using 16 word long bursts. When the FIFO 16 is full (*e.g.*, 40 to 64, 32 bit words), the current burst is completed, and no further data is requested. When FIFO 16 signals that it has room for a burst again, the image reading process from the frame buffer continues.

For dual scan operation, the FIFO 16 is split in two and operates with a separate FULL indicator for each half. In this mode, the FULL signal and a DS_FULL indicator (not shown) trigger from 12 to 32 words. For dual and single scan displays, information for the upper left corner of the display begins at a word address stored in a VIDSCRNPAGE register (not shown). For a dual scan display, information from the upper left corner of the lower half of the display begins at the word address stored in a

VIDSCRNHPG register (not shown). The VIDSCRPAGE and VIDSCRNHPG registers are used to pre-load address counters at the beginning of a video frame. The VILOSATI 14 continues to service the video FIFO 16 until it has transferred an entire screen image (*e.g.*, a frame) from memory. The size of the screen image is controlled by the values stored in a SCRNLINES register and a LINELENGTH register (not shown). The SCRNLINES register value defines the total number of displayed (active) lines for the video frame. The LINELENGTH register defines the number of words for each displayed (active) video line. A separate register, VLINESTEP (not shown), defines the word offset in memory between the beginning of each line and the next line. Setting the VLINESTEP value larger than the LINELENGTH value provides the capability for image panning.

The video FIFO 16 is used to buffer video data transferred from the frame buffer memory (*e.g.*, of frame buffer 68 of Fig. 2A) to the video output system without stalling the video data stream of the raster engine 2. The FIFO 16 comprises a dual port RAM 32 with input and output address index counters 34 and 36, respectively, and a control logic system 38 to operate as a FIFO memory. The input data bus width to the FIFO 16 is 32 bits. During dual scan mode, wherein the display requires scan out of the bottom and top half of the screen at the same time, top half (or bottom half) information is stored in every other FIFO location. In progressive scan mode wherein video data is scanned out as a single progressive image, the FIFO data is stored sequentially. The FIFO output data bus is 64 bits wide and can output even and odd words on both the upper and lower half of the bus. Writes to the FIFO 16 advance the input index counter 34, while reads from the FIFO 16 advance the output index counter 36. The input and output counters 34 and 36 are compared to generate the FULL and DS_FULL output controls to the VILOSATI 14. The N_CLR signal resets both the input and output index counters 34 and 36 to 0, for example, at the end of a video frame.

The control logic 38 in the FIFO system 16 includes an underflow detection and indication system which operates to detect an underflow of the FIFO 16 (*e.g.*, dual port RAM 32) and/or a near underflow condition therein, and to provide the Underflow_INT signal according to the detected underflow condition. The underflow system of the FIFO control logic 38 may include, for example, comparison logic for comparing the values of in and out counters 34 and 36, respectively, and for making a determination of whether an underflow condition exists or is anticipated. The Underflow_INT indication may be

advantageously provided to a host processor (*e.g.*, CPU 62 of Fig. 2A) whereby methods to balance bus loading or to limit burst sizes may be applied by the host processor. This feature is particularly advantageous where the raster engine interface with the frame buffer memory is via a bus isolated from that of the host processor. In this situation, the host may not be able to independently detect or sense bus loading conditions resulting in a starving raster engine. Thus, the invention provides for early indication to the host processor, whereby elimination or reduction in raster engine underflow conditions may be achieved.

Referring also to Fig. 2B, the pixel reconstruction system of the raster engine 2 includes a pixel multiplexer 18 and pipe-line registers (not shown), wherein the pixel multiplexer 18 is operative to 'unpack' the video pixels stored in the dual port RAM 32 of the video FIFO 16. The stored FIFO words (*e.g.*, 32 bit words in the dual port RAM 32) may be transferred 2 at a time across a 64 bit bus 33. The multiplexer 18 selects a single pixel to go on the 24 bit output bus based on the value set in a PIXELMODE register (e.g., in compare and register logic 4), as illustrated and described in greater detail hereinafter. The pixel multiplexer 18 is controlled by a pixel counter (not shown) that also increments based on the PIXELMODE register value.

The amount and frequency of data read from the FIFO 16 is dependent on the number of bits per pixel. For example, in an 8 bpp configuration, the 64 bit FIFO output is changed for every eight pixels. In dual scan mode, the upper 32 bits and lower 32 bits are read out in parallel and upper half screen and lower half screen pixels are unpacked and loaded into the video stream sequentially. The format of the video data in the frame buffer 68 may vary. For example, the data obtained by the dual port RAM 32 from the frame buffer 68 may comprise 4 bpp (bits per pixel), 8 bpp, 16 bpp 555 mode, 16 bpp 565 mode, 24 bpp mode or 32 bpp data formats. The pixel multiplexer 18 selects appropriate pixel data from the dual port RAM 32 according to a selected display mode, and accordingly provides the selected pixel data to match an output format required by the selected display type. The raster engine 2 thereby provides for selective remapping of the pixel data from the frame buffer format to a format appropriate for interfacing to a selected display device type, without requiring rerouting of signals outside of the raster engine. This remapping feature is provided via one or more user programmable control registers, which may be included within the compare and register logic 4 as illustrated in Fig. 2B, or which may reside elsewhere in the raster engine 2.

Bounded Video Signature Analyzer

Referring now to Fig. 3, the exemplary bounded video output signature analyzer 30 is illustrated having control registers 100 accessible to a host processor in the system (*e.g.*, system 60 of Fig. 2A) via an address bus 102 and a data bus 104 (*e.g.*, collectively system bus 66 of system 60), and further comprising a linear feedback shift register (LFSR) 106 receiving control signals 108 and parallel video data 110 from control registers 100, and providing video signature data 112 to a video output signature result register 114. Registers 100 may be, for example, included within the compare and register logic 4 of raster engine 2 in Fig. 1, and receive video data 116 from the mux 40 of the raster engine 2. Signature analyzer 30 may be used for built in self testing of reference images to ensure proper operation of the entire video system and data path. In addition, the signature analyzer 30 is operative to perform selective analysis of a portion of the video data from the raster engine 2. The bounded video signature analyzer 30 thus may perform signature analysis on one or more selected portions of the video data, in order to allow testing of video screen images having features which change over time (*e.g.*, clocks, date indications, and the like). The video timing section (*e.g.*, counters 28 and compare and register logic 4) of the exemplary raster engine 2 provides enable and clear control signals that determine the area of the output image that is used for the signature analysis calculation and at what time the next signature starts / last value is stored.

Referring also to Fig. 4, the video analyzer LFSR 106 is illustrated having parallel inputs input0 through input23 for incoming video data to be analyzed. Timing control signals are also fed into the LFSR 106 as parallel data to be analyzed. Each parallel input into the video signature analyzer LFSR 106 may be separately enabled in the control registers 100. Result storage register 114 receives a signature value from the LFSR 106 which is unique to the input video data 110, and may be read via the host computer system (*e.g.*, system 60 of Fig. 2A). For example, a new signature is calculated once per frame and stored based on a programmed signature clear location. During grayscale operation, the signature may be automatically taken over a 12 frame or other interval.

Depending on the refresh frequency of the display device 72, this could be a significant time interval. For example, the analyzer may have a calculation interval of 500 ms or more before updating the signature value. In addition, the signature analyzer

LFSR 106 includes a logical inversion 118 in the feedback chain, whereby a non-zero signature output is provided by LFSR 106 in response to zero parallel input data 110 from control registers 100. Thus, for a zero seed value and null inputs, a signature is still generated based on the number of clock pulses.

The integration of the signature analyzer 30 with the raster engine 2, allows the raster engine 2 to be tested after shipment to an end user or retailer, and further enables self-testing initiated via the control registers 100 by a user and/or an application programming running on a host computer system (*e.g.*, system 60). This integration provides significant advantages over conventional video signature analyzers and video controllers where a separate signature analyzer had to be connected to a raster engine to perform such signature analysis.

The signature analyzer 30, moreover, is bounded. The analyzer 30 may thus be programmed (*e.g.*, via control registers 100) to analyze a portion of a video screen data set, whereby selective avoidance of certain display areas may be achieved. Referring also to Fig. 5, an exemplary display screen 120 is illustrated having a clock image 122 displayed thereon. Thus, where it is known that the clock image 122 changes over time, the signature analyzer 30 may be adapted to selectively analyze one or more regions REGION 1 and/or REGION 2 in the display 120. Thus, the signature analyzer 30 may first analyze the video data between display locations (X1, Y1), and (X2, Y2) to obtain a signature for REGION 1, and subsequently analyze the video data between locations (X3, Y3) and (X4, Y4) to generate a signature for REGION 2. This capability allows successful signature analysis of the majority of the display 120 by comparison to known good signature information, without experiencing false indications of failure due to the changing nature of the clock image 122, which false indications were common in prior non-bounded signature analyzers.

Referring also to Figs. 6A through 6E, exemplary control registers SIGVAL 130, SIGCTL 132, VSIGSTRTSTOP 134, HSIGSTRTSTOP 136, and SIGCLR 138 are illustrated. The registers 130, 132, 134, 136, and 138 may be included within control registers 100 of Fig. 3. SIGVAL 130 is a video output signature result value register (*e.g.*, register 114 of Fig. 3), having reserved bits RSVD, and SIGVAL[15:0] bits. The read only SIGVAL value is the 16 bit result of the video output signature. This value may be updated once per frame based on the SIGCLR location. During grayscale operation, the SIGVAL register may be updated once every 12 frames. The SIGCTL

register 132 of Fig. 6B is a video output signature control register, having the following bit descriptions: EN: enable bit, which enables a linear feedback shift register; RSVD (reserved) bits; SPCLK bit which may be used to enable the SPCLK output for calculation in the video signature; BRIGHT bit used to enable the BRIGHTNESS control output for calculation in the video signature; a CLKEN bit used to enable the CLKEN control for calculation in the video signature; a HSYNC bit used to enable the HSYNC output for calculation in the video signature; a VSYNC bit is used to enable the VSYNC output for calculation in the video signature; and PEN[23:0] bits, which may be used to enable individual pixel bits for calculation in the video signature.

The SIGSTRTSTOP register 134 is a vertical signature bounds start/stop register, having reserved bits RSVD and STOP[10:0] bits to provide a value of a vertical down counter at which the VSIGEN signal goes inactive. This may be used to indicate the end of a signature calculation for a vertical frame. VSIGEN may be an internal block signal. The SIG_ENABLE control to the video signature analyzer may be enabled by the logical AND of VSIGEN and HSIGEN. In addition, the SIGSTRTSTOP register 134 further includes STRT[10:0] bits which indicate a value of the vertical down counter at which the VSIGEN signal becomes active. This may indicate the beginning of the signature calculation for the vertical frame. VSIGEN is an internal block signal. The SIG_ENABLE control to the video signature analyzer may be enabled by the logical AND of VSIGEN and HSIGEN.

The HSIGSTRTSTOP register 136 is a horizontal signature bounds start/stop register, having reserved bits RSVD and STOP[10:0] bits which indicate a value of the horizontal down counter at which the HSIGEN signal goes inactive, indicating the end of the signature calculation for a horizontal line. HSIGEN is an internal block signal. The SIG_ENABLE control to the video signature analyzer may be enabled by the logical AND of VSIGEN and HSIGEN. Register 136 further comprises STRT[10:0] bits indicating a value of the horizontal down counter at which the HSIGEN signal becomes active. This indicates the beginning of the signature calculation for a horizontal line. HSIGEN is an internal block signal. The SIG_ENABLE control to the video signature analyzer is enabled by the logical AND of VSIGEN and HSIGEN.

The SIGCLR register 138 is a signature clear location register having reserved bits RSVD and VCLR[10:0] bits which may indicate a value of the vertical down counter at which the VSIGCLR signal is active. This indicates the line for clearing the LFSR and

storing the result value for the vertical frame. VSIGCLR is an internal block signal. The SIG_CLR control to the video signature analyzer is generated by the logical AND of VSIGCLR and HSIGCLR. The SIGCLR control signal is also routed to an edge trigger capable interrupt on the interrupt controller for use as a programmable secondary REALITI interrupt output. Register 138 further comprises HCLR[10:0] bits which may indicate a value of the horizontal down counter at which the HSIGCLR signal is active. This indicates the specific horizontal pixel clock for clearing the LFSR and storing the result value within a horizontal line. HSIGCLR is an internal block signal. The SIG_CLR control to the video signature analyzer is generated by the logical AND of VSIGCLR and HSIGCLR. The SIGCLR control signal is also routed to an edge trigger capable interrupt on the interrupt controller for use as a programmable secondary REALITI interrupt output.

Hardware Cursor

The raster engine 2 further provides support for a hardware cursor, via the exemplary hardware cursor system 24 of Fig. 1. The hardware cursor system 24 is adapted to support dual as well as progressive scan display types according to a selected display mode, as described in greater detail hereinafter. Referring to Figs. 7A and 7B, a progressive scan display 150 is illustrated having a cursor image 152 displayed thereon. The cursor image 152 has a starting address 154 (*e.g.*, X and Y location), a vertical height 156, and a width 158, for example, where the height 156 and width 158 may be expressed in terms of lines and pixels, respectively. The hardware cursor system 124 is adapted to selectively overlay the cursor image 152 onto the display 150 in progressive scan mode. For a progressive scanned images, the system 24 is provided a starting address in memory for the cursor image 152, the X and Y location 154, the height 156 of the cursor in lines, and the width 158 of the cursor in pixels. A single line of the cursor image 152 is then loaded into the storage registers 100 of Fig. 1. As the display 150 is scanned, the system 24 waits for the appropriate X and Y location on the line and pixel counters (*e.g.*, horizontal and vertical counters 28 of Fig. 1), and then overlays the cursor data into the video stream via the mux 20.

Referring now to Figs. 8A and 8B, an exemplary dual scan display 160 is illustrated having adjacent first display portion 162 and second display portion 164, providing lower and upper halves of the display 160, respectively, and with a display

boundary 160A therebetween. The dual scan display 160 may be refreshed by scanning out the first and second display portions 162 and 164 at the same time in parallel. A cursor image 166 has a start address 168, a vertical height 170, and a width 172. The hardware cursor system 24 is adapted to selectively overlay the cursor image 166 onto one of the first and second portions 162 and/or 164, respectively of the display 160 in dual scan mode.

Referring also to Figs. 9A and 9B, the cursor image 166 is illustrated crossing the display boundary 160A, wherein a first portion 166A thereof is in the first or lower portion 162 of the display 160 having a first cursor portion height 170A, and wherein a second cursor portion 166B is in the second or upper display portion 164 having a second cursor portion height 170B. For dual scanned images, the hardware cursor system 24 is provided with the X and Y coordinates or location of where to begin inserting the cursor image 166 into the video stream, the address of where the first portion 166A of the cursor image 166 is to be overlayed, the Y location or coordinate of the second portion 166B of the cursor image 166 if applicable (*e.g.*, where the cursor image 166 crosses the display boundary 160A), the address at which to start looking for the next part of the cursor image 166 to be overlayed (*e.g.*, the second cursor portion 166B) after overlaying the last line of the cursor image first portion 166A, the first and second cursor portion heights 170A and 170B, respectively (if applicable), the cursor width 172, and whether the cursor image 166 is in the first display portion 162, the second display portion 164, or both (*e.g.*, cursor image 166 crosses the display boundary 160A).

The hardware cursor system 24 employs this information to overlay the cursor image 166 onto the display 160 by selectively inserting cursor image data into the video stream of the raster engine 2 via the mux 20. Initially, the first line of the first portion 166A of the cursor image 166 is loaded into one or more registers (*e.g.*, of compare and register logic 4) from the start address. As the display 160 is scanned, the cursor system 24 waits for the X and Y location on the horizontal and vertical counters 28, and overlays or inserts the appropriate cursor data into the video stream. In dual scan operation where the cursor image 166 appears only in one of the first and second display portions 162 and 164, respectively, the cursor image data is overlaid in the appropriate display portion. This process continues until all the cursor image data lines have been inserted into the video stream via the mux 20. If the cursor is entirely in one of the display portions 162 or 164, this completes the cursor image overlay until the next video image frame.

Where the cursor image 166 crosses the display boundary 160A, the hardware cursor system 24 jumps to the address location for the second cursor portion 166B, which is also known as the reset address. The first line of the second cursor portion 166B is then loaded into the storage buffer registers of compare and register logic 4. It will be appreciated that where the dual scanning simultaneously scans from top to bottom of each of the first (lower) portion 162 and the second (upper) portion 164 of the display 160, that the first (lower) cursor portion 166A will be overlayed into the video stream for the first (lower) display portion 162 prior to the second (upper) cursor portion 166B being overlayed into the video stream for the second (upper) display portion 164, although the invention contemplates other scanning methodologies. The system 24 then waits for the same X and the second Y location in the line and pixel counters (*e.g.*, via cursor output counters 56, compare and register logic 4, and horizontal and vertical counters 28). At the appropriate counter values, the cursor line buffer 58 overlays the second cursor portion 166B into the video stream for the second (upper) display portion 160B via the mux 20 until the second cursor portion 166B has been completely overlayed (*e.g.*, according to the height 170B of the second cursor portion 166B).

In this fashion, fast hardware cursor overlaying is provided for progressive as well as dual scanned display types according to a selected display type. The invention thus provides significant reduction in the processing resource overhead associated with conventional software cursor overlay techniques, and programmatically supports a variety of disparate display and cursor types. For example, the cursor image size may be adjustable to 16, 32, 48, or 64 pixels wide by up to 64 pixels in height, and may be stored anywhere in memory as a 2 bpp.

The image pixel information implies transparent, inverted, cursor color 1, or cursor color 2. The cursor hardware system 24 may be supplied an image starting address, 2 cursor colors, an X and Y screen location, and a cursor size. Using this information, the raster engine 2 overlays the cursor in the output video stream. Bottom and right edge clipping may also be performed by the raster engine hardware 24. The bus mastering interface 50 to an AMBA bus allows the hardware cursor image to be stored anywhere in host system memory (*e.g.*, memory 64 of Fig. 2A). Software provides a location start, reset, size, x & y position, and two cursor colors. The system 24 loads a line at a time from memory and multiplexes the video stream data based on the cursor values. The X & Y locations are compared to the horizontal and vertical counters (*e.g.*,

counters 28 of raster engine 2) and trigger the state machine 54 to enable the cursor output overlay via the cursor line buffer 58 and the mux 20.

The invention further comprises a method of overlaying a cursor image onto a dual scan display. Referring to Fig. 10, an exemplary method 180 is illustrated for each frame beginning at step 182. In dual scan display mode, decision step 184 determines whether the cursor image (*e.g.*, image 166 of Fig. 9A) crosses the display boundary (*e.g.*, display boundary 160A of display 160). If not, decision step 186 determines whether the cursor image is in the first display portion (*e.g.*, first display portion 162). If so, the cursor image is overlayed onto the first display portion at step 188. If not, the cursor image is overlayed onto the second display portion (*e.g.*, second display portion 164) at step 190. Where the cursor image crosses the display boundary at step, 184, the method 180 proceeds to step 192 where the first and second portions of the cursor are determined (*e.g.*, first and second cursor portions 166A and 166B of Fig. 9A). Thereafter, the first cursor portion is overlayed onto the first display portion at step 194, after which the second cursor portion is overlayed onto the second display portion at step 196. Once the cursor image has been thus overlayed onto the dual scanned display, the method 180 ends at step 198, until the next frame is to be scanned out.

Referring now to Figs. 11A through 11G, various registers operatively associated with the hardware cursor system 24 are illustrated and described hereinafter. It will be appreciated that the registers of Figs. 11A through 11G may be included in the compare and register logic 4 of the exemplary raster engine 2 in Fig. 1, or alternatively may be located elsewhere in the raster engine 2. In Fig. 11A, a CURSOR_ADR_START register 200 is illustrated. This register 200 is a cursor image address start register having reserved bits RSVD and ADR[31:2] bits indicating the beginning word location of the part of the cursor image to be displayed first. The image is 2 bits per pixel, and may be stored linearly. The amount of storage space is dependent on the width and height of the cursor. Reset is the beginning word location of the part of the cursor which will be displayed next after reaching the last line of the cursor. These locations are used for dual scan display of cursor information. If the cursor is totally in the upper half or lower half of the screen, the Start and Reset locations may be the same. Otherwise the cursor may be overlaid on the video information at the start address, and when the dual scan height counter generates a carry, will jump to the reset value. The cursor then continues to be overlaid when the Y location is reached, and will jump to the start address value when the

height counter for the upper half generates a carry. Offsetting the start value and changing the width of the cursor to be different from the cursor step value allows the appropriate 48, 32, or 16 pixels of a larger cursor to be displayed only. Furthermore, offsetting the starting X location off of the left edge of the screen allows pixel placement of the cursor off of the screen edge.

In Fig. 11B, a CURSOR_ADR_RESET register 202 is illustrated, having reserved bits RSVD and ADR[31:2] bits indicating the beginning word location of the part of the cursor which may be displayed next after reaching the last line of the cursor. Both start and reset locations are employed for dual scan display of cursor information. If the cursor is totally in the upper half or lower half of the screen, the Start and Reset locations may be the same. Otherwise (the cursor image crosses the display boundary) the cursor will be overlaid on the video information beginning at the start address, and when the dual scan height counter generates a carry, will jump to the reset value. The cursor will then continue to be overlaid when the Y location is reached, and will jump to the start address value when the height counter for the upper half (*e.g.*, the second display portion) generates a carry. Offsetting the reset value and changing the width of the cursor to be different from the cursor step value allows the appropriate 48, 32, or 16 pixels of a larger cursor to be displayed only. Furthermore, offsetting the reset X location off of the left edge of the screen will allow pixel placement of the cursor off of the screen edge.

A CURSORSIZE register 204 is illustrated in Fig. 11C for setting the cursor height, width, and step size, having reserved bits RSVD and DLNS[5:0] (dual scan lower half lines) bits which may be set to the number of cursor lines displayed in the lower half of the screen in dual scan mode. Register 204 further comprises CSTEP[1:0] cursor step size bits, which control the counter step size for the width of the cursor image. For example, the following cursor step sizes are possible according to the CSTEP bits: 00 = step by 1 word or 16 pixels at a time, 01 = step by 2 words or 32 pixels at a time, 10 = step by 3 words or 48 pixels at a time; and 11 = step by 4 words or 64 pixels at a time. The register 204 further comprises CLINS[5:0]: cursor line bits, which control height in lines of the cursor image. The value may be set, for example, to the number of lines minus 1. In a dual scan mode this may be set to the number of cursor lines displayed in the top half of the screen. Also included in register 204 are CWID[1:0]: cursor width bits, which control the displayed word width (minus 1) of the cursor image, which may have the following values: 00 = display 1 word or 16 pixels; 01 = display 2 words or 32

pixels; 10 = display 3 words or 48 pixels; or 11 = display 4 words or 64 pixels.

In Fig. 11D, the CURSORCOLOR1, CURSORCOLOR2, CURSORBLINK1, and CURSORBLINK2 registers 206 are illustrated for defining the color of the displayed cursor image. The registers have the following bit definitions: RSVD: Reserved; COLOR[23:0]: Image color inserted directly in the video pipeline, which overlays all other colors when cursor enabled, and may not go through LUT. (*e.g.*, look up table 10). The 2 bit per pixel stored cursor image bits may, for example, be displayed as follows: 00 = transparent; 01 = invert video stream; 10 = CURSORCOLOR1 during no blink, CURSORBLINK1 during blink; and 11 = CURSORCOLOR2 during no blink, CURSORBLINK2 during blink.

Referring to Fig. 11E, a CURSORXYLOC register 208 is illustrated for defining the X and Y cursor location, which includes reserved bits RSVD and YLOC[10:0] bits which control the starting vertical Y location of the cursor image. The value is used to compare to the vertical line counter and may be set by software to be between the active start and active stop vertical line values. The cursor hardware 24 may clip the cursor at the bottom of the screen. The new location value may not be used until the next frame to prevent cursor distortion. Also included in the register 208 is a CEN bit, which may be used to enable the hardware to insert the defined cursor into the image output video stream. For example, when active, data from a location defined by the CURSORADR register may be combined with the output video stream. Thus, the CEN bit may have the following values: 0 = hardware cursor not activated; and 1 = hardware cursor activated. During dual scan mode this bit may be used to indicate that some or all of the cursor is located on the upper half of the screen. The XLOC[10:0]: bits control the starting horizontal X location of the cursor image. The value may be used to compare to the horizontal pixel counter and may be set by software to be between the active start and active stop horizontal pixel values. The cursor hardware may clip the cursor at the right edge of the screen. This value may also be used to control the starting location for the cursor image on the upper half of the screen during dual scan mode. The new location value may not be used until the next frame to prevent cursor distortion.

In Fig. 11F, a CURSOR_DHSCAN_LH_YLOC register 210 is illustrated for indicating the X and Y cursor location. This register 210 includes reserved bits RSVD, a CLHEN bit (cursor lower half enable) indicating that some or all of the cursor is located on the lower half of the screen, and YLOC[10:0]bits, wherein during dual scan display

mode, the YLOC[10:0] value controls the starting vertical Y location on the lower half of the screen for the cursor image. The value may be used to compare to the vertical line counter and may be set by software to be between the active start and active stop vertical line values. The cursor hardware may clip the cursor at the bottom of the screen. The new location value may not be used until the next frame to prevent cursor distortion. In Fig. 11G, a CURSORBLINK register 212 is illustrated, which may be used to control the blink rate for the cursor image. CURSORBLINK register 212 includes reserved bits RSVD and an EN (hardware cursor blinking enable) bit used to enable blinking for CURSORCOLOR1 and CURSORCOLOR2 to CURSORBLINK1 and CURSORBLINK2 registers (206) respectively. This bit may also enable the cursor blink rate counter, according to the following values: 0 = hardware cursor blinking not activated, and 1 = hardware cursor blinking activated. Register 212 further comprises RATE[7:0] bits. The value of the RATE bits may be used to control the number of video frames that occur before switching between CURSORCOLOR1 or CURSORCOLOR2 and CURSORBLINK1 or CURSORBLINK2 registers (206) respectively. An on/off cursor blink cycle may be controlled by the following equation: Blink Cycle = 2 x (1/VXTAL2) x HCLKSTOTAL x VLINESTOTAL x (255 - BLINKRATE). This pertains to a 50% duty cycle blink rate, however other duty cycle blink rates may be attained by using an appropriate count value and comparison value.

In the above registers 200-212, Start is the beginning word location of the part of the cursor image to be displayed first. The image may be 2 bits per pixel, and may be stored linearly. The amount of storage space may depend on the width and height of the cursor. The two bits correspond to show screen image (transparent), invert screen image, display color1, and display color2. Reset is the beginning word location of the part of the cursor which will be displayed next after reaching the last line of the cursor. These locations may be advantageously employed for dual scan display of cursor information. For example, if the cursor is totally in the upper half or lower half of the screen, the Start and Reset locations may be the same. Otherwise (the cursor crosses the display boundary), the cursor may start being overlaid on the video information at the start address, and when the dual scan height counter generates a carry, may jump to the reset value. The cursor may then continue to be overlaid when the Y location is reached, and may jump to the start address value when the height counter for the upper half generates a carry.

Offsetting these values and changing the width of the cursor to be different from the cursor step value allows the right 48, 32, or 16 pixels of a larger cursor to be displayed. In addition, offsetting the starting X location off of the left edge of the screen may allow pixel placement of the cursor off of the screen edge. The size may be specified as a width adjustable to 16, 32, 48, or 64 pixels, a height in lines up to 64 pixels (*e.g.*, controls the top half of the screen only in dual scan mode), a step size for number of words in a cursor line up to 4, and a height of up to 64 lines on the bottom half of the screen used in dual scan mode. The Y location value may control the starting vertical Y location of the cursor image. The value may be used to compare to the vertical line counter and may be set by software to be between the active start and active stop vertical line values. The cursor hardware 24 may clip the cursor at the bottom of the screen. The new Y location value may not be used until the next frame to prevent cursor distortion.

The X location value controls the starting horizontal X location of the cursor image. The value is used to compare to the horizontal pixel counter (*e.g.*, horizontal and vertical counters 28) and may be set by software to be between the active start and active stop horizontal pixel values. The cursor hardware 24 may clip the cursor at the right edge of the screen. This value may be also used to control the starting location for the cursor image on the upper half of the screen during dual scan mode. The new X location value may not be used until the next frame to prevent cursor distortion. During dual scan display mode, the lower half Y value controls the starting vertical Y location on the lower half of the screen for the cursor image. The value may be used to compare to the vertical line counter and may be set by software to be between the active start and active stop vertical line values. The cursor hardware may clip the cursor at the bottom of the screen. The new location value may not be used until the next frame to prevent cursor distortion. The hardware cursor system 24 further includes a separate blinking function, wherein the rate may be a 50% or alternately other duty cycle programmable number of vertical frame intervals. For example, when a blink frame is active, the mux 20 may switch in 24 bit BLINKCOLOR1 and BLINKCOLOR2 values for CURSORCOLOR1, and CURSORCOLOR2, respectively.

Multiple Color Depth Interface

Referring now to Figs. 1 and 12, the raster engine 2 comprises the dual 256 x 24-bit SRAM 10 used as a pixel color look up table (LUT). One LUT may be inserted in the

video pipeline, while the other may be accessible by the system processor via the AHB bus. Writing a control bit selects which LUT is in the video pipeline and which is accessible via the bus. The dual LUT 10 may be memory mapped with respect to a raster engine base address and accessible from the AHB bus, one LUT at a time. During active video display, an LUT switch command may be synchronized to the beginning of the next vertical frame. The status of actual switch occurrence may be monitored on an LUTCONT.SSTAT bit (not shown) in the registers 4, which may be polled. Alternatively, the frame interrupt may be enabled and used to time the switching. Each table in the dual LUT 10 may be used for 4 bpp and 8 bpp modes and may be beneficial to bypass for 16 bpp and 24 bpp modes since a reduction in the number of simultaneously available colors would result. Control for whether or not the dual LUT 10 is used or bypassed altogether in the video pipeline is performed by configuring a PIXELMODE register color definition value, as illustrated and described in greater detail hereinafter. The PIXELMODE and other registers may thus be programmed by a user or by an application program to select and implement display modes for a variety of disparate display types.

The color RGB mux 20 is adapted to select appropriate pixel data and to provide the selected data to the appropriate video output stream. The mux 20 selects pixel data from the LUT 10, the grayscale generator 12, the hardware cursor logic 24, or directly from the pipeline after the blink logic system 8 according to the selected display mode. Mux 20 formats data for the pixel shift logic 22, a color digital to analog converter (DAC) 6, and/or for the YCRCB interface 26. The formatted video output data may be provided to a display device (not shown) via the output mux 40 together with data and clock buffers 42 and 44, respectively. The selected display mode is programmable to determine the operating mode for the mux 20, the pixel shift logic system 22, the blink logic system 8, LUT 10, and the grayscale generator 12, as well as for the signature analyzer 30 and hardware cursor system 24, as described above. For example, the mode of operation for the mux 20 may be set by the value of the PIXELMODE register. Accordingly, the mux 20 selects video data from the grayscale generator 12, from the LUT 10, or from the video pipeline after the blink logic 8 according to the selected display mode.

When the hardware cursor 24 is enabled, cursor data values may be injected into the pipeline via the mux 20, or alternatively, the primary incoming video data may be

inverted. When in 16-bit 555 or 565 data display modes, the pixel data may be reformatted to fit into a 24-bit bus. This may include copying the MSBs for the data into one or more unused LSBs of the bus to allow full color intensity range. Once selected and formatted, output data is provided by the mux 20 to the pixel shift logic system 22 , the YCrCb encoder 26, and/or the DAC 6.

The pixel shifting logic system 22 allows for reduced external data and clock rates by performing multiple pixel transfers in parallel. The output can be programmatically adapted (*e.g.*, via the compare and register logic 4) to transfer a single pixel per clock up to 24 bits wide, a single 24-bit or 16-bit pixel mapped to a single 18 bit pixel output per clock (*e.g.*, triple 6 RGB on 18 active data lines), 2 pixels per clock up to 9 bits wide each (18 pixel data lines active), 4 pixels per clock up to 4 bits wide each (16 pixel data lines active), or 8 pixels per clock up to 2 bits wide each (16 pixel data lines active). The pixel shifting logic system 22 may also be programmed to output 2 and 2/3, 3 bit pixels on the lower 8 bits of the bus per pixel clock or to operate in a dual scan 2 and 2/3 pixel mode putting 2 and 2/3 pixels from the upper and lower halves of the screen on the lower 8 bits of the bus and the next 8 bits of the bus per clock respectively. In dual scan mode, every other pixel in the pipeline may be from the other half of the display. Dual scan mode support may thus be provided for various formats, including 1 upper/1 lower pixel, 2 upper/2 lower pixels, and 4 upper/4 lower pixels corresponding to the 2 pixels per clock, 4 pixels per clock and 8 pixels per clock modes.

Referring also to Figs. 13A through 13C, the compare and register logic may further comprise a PIXELMODE register 230, a PARLLIFOUT register 232, and a PARLLIFIN register 234. The PIXELMODE register 230 is adapted to indicate a selected display mode for the operation of the raster engine 2, and includes reserved bits RSVD and a DSCAN (dual scan enable) bit for servicing dual scanned displays. When active, data from two locations in memory (top and bottom halves of the screen) may be piped through the video pipeline every other pixel. The output pixel shift logic system 22 accordingly drives the top and bottom half screen data at the same time. This mode may be employed, for example, in association with passive matrix LCD screens that require both halves of the screen to be scanned out at the same time, or alternatively, may be used to drive two separate screens with different data. The values for the DSCAN bit may include: 0 = half page mode not activated, and 1 = half page mode activated.

The PIXELMODE register 230 further comprises C[3:0]: color mode definition

bits having values indicating a selected color mode according to the following table:

C3	C2	C1	C0	Color Mode
X	0	0	0	Use LUT Data
X	1	0	0	Triple 8 bits per channel
X	1	0	1	16-bit 565 color mode
X	1	1	0	16-bit 555 color mode
1	X	X	X	Grayscale Palette Enabled

In addition, PIXELMODE register 230 includes M[3:0]: blink mode definition bits, having values which indicate a selected blink mode according to the following table:

M3	M2	M1	M0	Blink Mode
0	0	0	0	Blink Mode Disabled
0	0	0	1	Pixels ANDed with Blink Mask
0	0	1	0	Pixels ORed with Blink Mask
0	0	1	1	XORed with Blink Mask
0	1	0	0	Blink to background register Value
0	1	0	1	Blink to offset color single value mode
0	1	1	0	Blink to offset color 888 mode (555,565)
0	1	1	1	Undefined
1	1	0	0	Blink dimmer single value mode
1	1	0	1	Blink brighter single value mode
1	1	1	0	Blink dimmer 888 mode (555,565)
1	1	1	1	Blink brighter 888 mode (555,565)

PIXELMODE register 230 further comprises S[2:0]: output shift mode bits, having values indicating a selected shift mode according to the following table:

S2	S1	S0	Shift Mode
0	0	0	1 - pixel per pixel clock (up to 24 bits wide)
0	0	1	1 - 24-bit or 16-bit pixel mapped to 18 bits each pixel clock

0	1	0	2 - pixels per shift clock (up to 9 bits wide each)
0	1	1	4 - pixels per shift clock (up to 4 bits wide each)
1	0	0	8 - pixels per shift clock (up to 2 bits wide each)
1	0	1	2 2/3 3-bit pixels per clock over 8 bit bus
1	1	0	Dual Scan 2 2/3 3-bit pixels per clock over 8 bit bus
1	1	1	Undefined - Defaults to 1 - pixel per pixel clock

The PIXELMODE register 230 also comprises pixel mode bits P[2:0]: having values indicating a selected number of bits per pixel scanned out by the raster engine 2, according to the following table:

P2	P1	P0	Pixel Mode
0	0	0	pixel multiplexer disabled
0	0	1	4 bit per pixel
0	1	0	8 bits per pixel
0	1	1	do not use
1	0	0	16 bits per pixel
1	0	1	do not use
1	1	0	24 bits per pixel
1	1	1	32 bits per pixel

Referring also to Fig. 13B, the compare and register logic 4 of the raster engine 2 further comprises a PARLLIFOUT register 232 (*e.g.*, parallel interface output control register) having a RD bit for controlling reads of the register 232. When writing to register 232, a >0= in this bit location will initiate a parallel interface write cycle and a >1= in this bit location initiates a parallel interface read cycle. In addition, register 232 includes DAT[7:0] bits, adapted to indicate the data output on the parallel interface during a write cycle. The DAT[7:0] bits may be driven onto C/VSYNCn, HSYNCn, BLANKn, P[17]/AC, and P[3:0] lines respectively.

In Fig. 13C, a PARLLIFIN register 234 (parallel interface control register) is illustrated, having reserved bits RSVD and ESTRT[3:0] (E enable signal start value) bits, which indicate the value of the parallel interface counter where the E enable signal becomes active (high). The data buffer enable also becomes active at the same time as

the E enable signal during a write cycle. The E enable signal becomes inactive just before the counter changes to 0, while the data is driven throughout the 0 count. This allows data to be driven active for one additional clock cycle to provide hold time to the display when writing. Register 234 further includes CNT[3:0] counter preload value bits adapted to indicate a value loaded into a parallel interface down counter. When a write or read command is issued by writing to register 234, the counter begins to count down from this value.

Additional IO lines (not shown) may be used to provide a read vs. write status indication, a data vs. instruction indication, and any address or chip select control signals. Raster engine 2 may thus provide a direct display command interface for interfacing a host processor (*e.g.*, CPU 62) of Fig. 2A with a low cost display, such as an LCD, having a command interface. The difference between the CNT[3:0] value and the ESTRT[3:0] value operates to ensure setup timing for write data and IO signals to an integrated display module before the rising edge of the E enable signal. In addition, the register 234 comprises DAT[7:0] bits, which indicate the data input on the parallel interface during a read cycle. The DAT[7:0] bits may be loaded into the LSB of this register from C/VSYNCn, HSYNCn, BLANKn, P[17]/AC, and P[3:0] lines, respectively, on the falling edge of the E interface enable control signal. The direct display command interface may be employed, for example, in interfacing a display module having a built-in processor which receives command words from the raster engine 2, rather than rasterized data. This feature enables a high speed host processor (*e.g.*, CPU 62 of Fig. 2A) to provide display commands to such a low-end display module (which typically operates at a much lower speed than does the host processor) via the raster engine, which provides appropriate timing and enable signaling to interface with the display module. Thus, the provision in the raster engine 2 of direct display command operating modes allows a processor to easily and efficiently provide commands to such a display module.

Referring also to Figs. 14A and 14B, a table 236 indicates various exemplary output transfer modes which are programmable in the raster engine 2 via the PIXELMODE register 230. As can be seen in the table 236, the selection of a shift mode (*e.g.*, via the S[2:0]: output shift mode bits) and the color mode (*e.g.*, via the C[3:0]: color mode definition bits) provides programmable support for a plurality of different display types having various video data output display modes. For example, the selected shift mode and color mode may be used to support various display modes, including: single

pixel per clock up to 24 bits wide; single 16 bit 565 pixel per clock; single 16 bit 555 pixel per clock; single 24 bit pixel on 18 lines; single 16 bit 565 pixel on 18 lines; single 16 bit 555 pixel on 18 lines; 2 pixels per clock; 4 pixels per clock; 8 pixels per shift clock; 2 2/3 pixels per clock; and dual 2 2/3 pixels per clock. Thus, the raster engine 2 provides the capability of outputting a plurality of pixels in a single shift clock.

The raster engine 2 may thus programmatically translate selected pixel data from a first format to a second format according to the selected display mode. As further indicated in the table 236, the raster engine may selectively translate video data between formats having disparate numbers of bits. For example, where the first format comprises more bits than does the second format, the raster engine 2 may selectively interpolate between a portion of the selected pixel data in the first format and generate a portion of the data in the second format (*e.g.*, via the pixel shift logic 22). This may be accomplished, for example, via performing a logical OR combination of at least two bits of the selected pixel data in the first format to generate a bit in the second format. This selective interpolation accomplishes a rounding which provides for maximum utilization of available colors, thus significantly improving color usage compared with simple truncation of unused bits.

As can be seen in table 236 of Figs. 14A and 14B, the raster engine provides a programmable interface to a plurality of disparate display device types. In this regard, the raster engine employs a universal routing scheme (*e.g.*, as illustrated in the table 236) whereby a variety of such disparate display types may be interfaced with a host computer (*e.g.*, CPU 62 of Fig. 2A). While prior raster engines required rerouting of output signals outside of the raster engine, no such rerouting is required in order to employ the raster engine 2. In addition, the raster engine 2 may be employed to interface with display devices using only four data bits, while still providing support for multiple video interface formats. In this regard, a control bit (not shown) is provided in the raster engine 2 which may be programmable via the PIXELMODE register 230 in order to invoke this operation as indicated in the table 236 (*e.g.*, P(13), P(9), P(5), and P(1)).

Programmable Hardware Blinking

Referring now to Figs. 1 and 15, the raster engine 2 further comprises the pixel blink logic system 8 adapted to blink pixels based on a selected blink mode. The pixel blink logic system 8 may be operatively associated with one or more control registers in

the compare and register logic 4 or elsewhere in the raster engine 2. Referring also to Figs. 16A through 16E, the number of video frames for a blink cycle may be controlled by a value in a BLINKRATE register 250, as described in greater detail hereinafter. The system 8 is further adapted to determine which pixels are designated as blinking pixels. Pixel blinking may be programmatically accomplished in several different ways, some of which may employ the look up table 10. This is done via the blink logic system 8 logically transforming the address into the look up table 10 based on whether the pixel is a blink pixel, and whether it is currently in the blink state, as well as a selected display mode. For example, a red blinking pixel may be set up to normally address location 0x11 in the look up table. When not in the blink state, the color output from this location would be red. In the blink state, the address could be logically modified to 0x21 via the blink logic system 8 according to the values in one or more control registers 4. The color stored at the 0x21 location could be green or black or whatever other color that it is desired for red in the blink state. For every pixel color, there may be a blinking version.

For LUT blinking, the address may be modified by using a masked AND/OR/XOR function according to a selected blink mode. A mask may be defined in a BLINKMASK register, as described in greater detail hereinafter with respect to Fig. 16B. Selection of whether the pixel data is ANDed, ORed, or XORed with the mask is set by the PIXELMODE register 230 of Fig. 13A. In another mode of blink operation, the blink function may be performed by logical or mathematical operations on the pixel data via the system 8. Such logical and/or mathematical operations may be programmed, for example, to implement blink to background, blink dimmer, blink brighter, or blink to offset blink modes by setting an appropriate PIXELMODE register value.

For example, when blink to background mode is enabled, the blink logic system 8 may selectively replace a blinking pixel with the value in a BG_OFFSET register, as illustrated and described in greater detail hereinafter with respect to Fig. 16E. Setting this register to the background screen color in this mode may cause an object to appear and disappear. Blink brighter and blink dimmer modes may also be achieved, wherein pixel data values may be shifted by one or more bit locations. For example, to blink brighter, the LSB may be dropped, the MSBs may be all shifted one bit lower, and the MSB may be set to a '1'. For blink dimmer, the LSB may be dropped, the MSBs may be all shifted one bit lower, and the MSB may be set to a '0'. Blink to offset may be accomplished by adding the value in the BG_OFFSET register to blinking pixels. The shifting and

offsetting can be programmed to be compatible with the selected pixel organization mode. Many different blinking modes are possible within the scope of the invention, whereby programmable hardware blinking of one or more pixels in a display may be accomplished.

A blinking pixel may be defined by a BLINKPATRN register and a PATTRNMASK register, as illustrated and described in greater detail hereinafter with respect to Figs. 16C and 16D. By using the PATTRNMASK register, either multiple or single bit planes may be used to specify blinking pixels. This allows the number of definable blinking pixels to range from all pixel combinations blinking to only one pixel that blinks. In addition, this feature allows the option of minimizing the number of lost colors by reducing the number of blinking colors, thus providing significant flexibility and advantages over conventional palette blinking techniques. The BLINKPATRN register may then be used to define the value of the PATTRNMASKed bits that should blink.

Referring now to Figs. 16A through 16E, several control registers are illustrated and described hereinafter, which are operatively associated with the blink logic system 8 of the raster engine 2. A BLINKRATE register 250, BLINKMASK register 252, BLINKPATRN register 254, PATTERNMASK register 256, and a BG_OFFSET register 258 may be employed in association with the system 8 in order to achieve the selective pixel blinking in accordance with the invention. The registers 250-258, moreover, may be included in the compare and register logic 4 of raster engine 2, or alternatively may be located elsewhere in the raster engine 2.

The number of video frames for a blink cycle may be controlled by a value in the BLINKRATE register 250 of Fig. 16A, which may comprise reserved bits RSVD, as well as RATE[7:0] bits. The value of the BLINKRATE register is programmable via the RATE bits to control the number of video frames that occur before the LUT addresses assigned to blink switch between masked and unmasked. Thus, an on/off blink cycle may be controlled according to the following equation: Blink Cycle = 2 x (1/VXTAL2) x HCLKSTOTAL x VLINESTOTAL x (255 - BLINKRATE), wherein the HCLKSTOTAL and VLINESTOTAL represent the value of counters (not shown) in the raster engine 2. This pertains to a 50% duty cycle blink rate, however other duty cycle blink rates may be attained by using a count value and a comparison value.

The BLINKMASK register 252 illustrated in Fig. 16B may comprise reserved bits

RSVD, along with mask bits MASK[23:0]. The value of the BLINKMASK register 252 may be ANDed, ORed, or XORed with a pixel data address into the look up table 10 defined as a blink pixel during a blink cycle. The programmable mask allows a blinking pixel to jump from the normal color definition location to a blink color location in the look up table 10 according to whether the pixel is in the blinking state or the non-blinking state. A logical AND operation may accordingly modify the LUT address by clearing bits, whereas an OR operation modifies the LUT address by setting bits, and an exclusive OR operation (XOR) modifies the LUT address by inverting bits.

Referring also to Fig. 16C, the BLINKPATRN register 254 defines a blink pattern for use by the blink logic system 8, which comprises reserved bits RSVD as well as pattern bits PATRN[23:0]. After being masked with the value of the PATTRNMASK register 256 described hereinafter, the PATRN value may be compared with pixel data bits (*e.g.*, bits 23-0) in order to determine when pipeline pixels are defined as blink pixels. Thus, the blink logic system 8 may be adapted to determine whether a pixel is a blinking pixel or not. In Fig.16D, the PATTERNMASK register 256 is illustrated, having reserved bits RSVD and pattern mask bits PMASK[23:0]. These bits PMASK[23:0] may be used to determine which PATTRN[23:0] bits of the BLINKPATRN register 254 are to be used to define pixels as blinking pixels. For example, the PMASK bits may have the following values: 0 = bit used for comparison, and 1 = bit not used for comparison.

Referring also to Fig. 16E, the BG_OFFSET register 258 is illustrated having reserved bits RSVD along with bits BGOFF[23:0] which may be used to set a blink background color or a blink offset value. The function of the BG_OFFSET register 258 may change based on the selected blink mode. For example, when the M[3:0] bits of the PIXELMODE register (*e.g.*, register 230 of Fig. 13A) are set to select a blink to background blink mode, the BG_OFFSET register 258 may be used by the blink logic system 8 to define a 24 bit color for the background. Alternatively, when the M[3:0] bits of the PIXELMODE register 230 are set to a blink to offset blink mode, the BG_OFFSET register 258 may be used by the blink logic system 8 to define a mathematical offset value for the blink color. In this regard, the format for the mathematical offset may be based on the selected display mode (*e.g.*, 888, 565, 555).

Grayscale Generator

As illustrated in Figs. 1, 17, and 18, the raster engine 2 further provides a programmable grayscale generator 12 adapted to provide grayscales for monochrome displays via one or more control registers, which may but need not be included within the compare and register logic 4. The grayscale generator 12 may be inserted in the video pipeline by the mux 20 according to a selected display mode. The grayscale generator 12 translates a 3 bit input to a single monochrome bit dithered output, thereby providing 8 shades of gray including black and white. The grayscale generator 12 may further comprise six 2-bit counters; FRAME_CNT3 270, FRAME_CNT4 272, VERT_CNT3 274, VERT_CNT4 276, HORZ_CNT3 278, and HORZ_CNT4 280 as illustrated in Fig. 18 and described in greater detail hereinafter.

A look up table or matrix in the grayscale generator 12 (or elsewhere in the raster engine 2, *e.g.*, in compare and register logic 4) may be programmed with values that define the on/off dithering operation for a pixel value based on value of one or more of the counters 270-280, as illustrated and described in greater detail hereinafter with respect to Figs. 20-30. A matrix size or dimension may be defined for each pixel value (*e.g.*, 0 through 7 for 3 bits). The matrix size may be from 3 horizontal rows x 3 vertical columns x 3 frames (*e.g.*, 3Hx3Vx3F) to 4H x 4V x 4F, or any combination of 3 or 4. It will be appreciated that wile the exemplary grayscale generator 12 provides for matrices varying from 3Hx3Vx3F to 4Hx4Vx4F, that the many different matrix sizes are possible, and are contemplated as being within the scope of the invention. The grayscale look up table is then filled in for each pixel with this matrix information. The grayscale generator 12 uses the programmed matrix to perform grayscaling according to the selected display mode, which is particularly advantageous when employed in association with low cost or monochrome displays.

Referring also to Fig. 19, a GRAYSCALE LUT register 282 may be provided in the raster engine and operatively associated with the grayscale generator 12. It will be noted that the register 282 may be included in the compare and register logic 4, or may be located elsewhere in the raster engine 2. GRAYSCALE LUT register 282 may be used to fill the matrix, and comprises reserved bits RSVD, as well as a FRAME bit, defining a frame counter selection for the current 3 bit pixel value wherein 0 = use FRAME_CNT3 and 1 = use FRAME_CNT4. In addition, the register 282 comprises a VERT bit defining a vertical counter selection for the current 3 bit pixel value wherein 0 = use VERT_CNT3 and 1 = use VERT_CNT4, as well as a HORZ bit. A horizontal counter selection may be

defined for the current 3 bit pixel value using the HORZ bit, wherein 0 = use HORZ_CNT3 and 1 = use HORZ_CNT4. In this manner, the matrix size may be programmed using the FRAME, VERT, and HORZ bits via the register 282.

The GRAYSCALE LUT register 282 further includes matrix position enable bits D[15:0]. These bits D[15:00] may be used to control/dither a monochrome data output according the to horizontal position, the vertical position, the frame, and the 3 bit incoming pixel definition. The grayscale matrix is thus fully programmable by a user or an application program to provide selective grayscaling according to a selected display mode for the raster engine 2. This allows the raster engine 2 to obtain pixel data from a frame buffer (*e.g.*, frame buffer 68 of Fig. 2A) and to generate grayscale formatted video data according to the selected display mode.

Referring now to Fig. 20, an exemplary grayscale matrix 300 is illustrated having a dimension 4Hx4Vx4F. The bit positions in the matrix 300 are illustrated corresponding to the GRAYSCALE LUT register 282 of Fig. 19. As an example of programming the grayscale matrix, Fig. 21 illustrates another exemplary grayscale matrix 302, wherein the grayscale matrix 300 is programmed for full on and full off operation. For example, where a pixel input value of zero (*e.g.*, 000 binary for three bit) is off, setting register addresses 0x80, 0xA0, 0xC0, and 0xE0 to all 0 ensures that a 0 pixel never turns on. Assuming that a pixel value of seven (*e.g.*, 111 binary) is full on, setting addresses 0x9C, 0xBC, 0xDC, and 0xFC to all 1, ensures that the value is always on. The values between full on and full off may be programmed according to any criteria, including the characteristics of a particular display type, for example, contrast, persistence, turn on time, turn off time, on/off duty cycle, and refresh rate.

To achieve different shades of gray, more values may be provided below half the luminance average, due to the higher sensitivity to luminance variations by the human eye at lower levels. Other considerations in programming the grayscale matrix include temporal distortion (*e.g.*, flickering), spatial distortion (*e.g.*, walking patterns), and spatial interference patterns. Referring now to Fig. 22, a fifty percent duty cycle 4Hx4Vx4F matrix 304 is graphically illustrated. This particular matrix definition in Fig. 22 may be subject to temporal distortion or flickering due to each pixel being turned on and turned off together.

Referring now to Fig 23, another exemplary fifty percent duty cycle 4Hx4Vx4F matrix 306 is illustrated. In order to avoid flickering, every other pixel may be turned on,

such that the human eye integrates the on and off pixels between two consecutive frames. The matrix definition of Fig. 23, however, may suffer from spatial interference, particularly wherein image displayed in this grayscale requires that every other column be activated (*e.g.*, a checkerboard pattern). Referring also to Figs. 24 and 25, this type of spatial interference may be minimized by mixing up the pattern sequence as illustrated in the 4Hx4Vx4F matrix 308. This pattern mixes two sets of adjacent pixels with sets of every other pixel. The matrix 308 may suffer from a walking pattern type of distortion, depending on the display type. Assuming that a three bit pattern representing the fifty percent duty cycle grayscale of Fig. 24 is 011 binary, the matrix 310 of Fig. 25 illustrates the programming of the grayscale matrix of the grayscale generator 12 for the pattern of Fig. 24.

Referring now to Fig. 26, another exemplary grayscale matrix 312 is illustrated with a 3Hx3Vx3F dimension. According to this exemplary grayscale dithering pattern, each cell in the matrix 312 is active for only one frame in any three frame sequence, thus achieving a thirty three percent duty cycle for each pixel. This 3Hx3Vx3F matrix 312 may also suffer from spatial distortion, since as the frame number progresses, the bit pattern in each row moves one pixel to the right. For example, diagonal lines in a displayed image using the grayscale matrix 312 may accordingly appear as though they are moving or walking to the right.

Turning now to Fig. 27, another exemplary 3Hx3Vx3F matrix 314 is illustrated which reduces the walking distortion potential of the matrix 312, via a slightly different dithering pattern. Assuming the 3 bit input pattern that represents the thirty three percent duty cycle grayscale of matrix 314 is 010 binary, the matrix may be programmed as illustrated in the register matrix 316 of Fig. 28. With the look up table or matrix 316 thus programmed into the control registers, the grayscale generator 12 may accordingly provide grayscaling in accordance therewith. Referring now to Figs. 29 and 30, non-symmetrical matrix sizes are further possible in accordance with the invention. An exemplary 4Hx3Vx3F matrix 318 is illustrated graphically in Fig. 29. Referring also to Fig. 30, and assuming that the three bit input pattern that represents a thirty three percent duty cycle grayscale is 010 binary, the programmed register matrix 320 further illustrates the programming of the grayscale matrix.

Referring now to Fig. 31, the raster engine 2 may be employed in a variety of systems having disparate display types and data formatting requirements. For example,

the table 350 illustrates several of the possible applications of the raster engine 2 with various display types. While the invention has been described herein in association with certain display types, it will be recognized that the invention is useful for other applications involving other display types not specifically illustrated and described herein. In addition the invention may be implemented as part of a system having other components and features.

Referring now to Fig. 32, a system 400 is illustrated in which various aspects of the present invention may be carried out. As illustrated and described above, the raster engine 2 may be employed in various computer systems (*e.g.*, system 60 of Fig. 2A). In addition, the raster engine 2 may be employed in other applications within the scope of the invention. For example, the raster engine 2 may be included within the system 400 of Fig. 32 as part of a video interface 402, wherein the system 400 may comprise a multi-function integrated circuit or chip having multiple components in addition to the video interface 402. The video interface 402 may be operatively connected to a bus 404 providing communications between various system components, as described hereinafter, including a processor 406.

The processor 406 may communicate via the bus 404 with various memory and peripheral components within the system 400. Included among these are a DRAM (dynamic random access memory) interface 414, an SRAM (static random access memory) and flash memory interface 416, a DMA (direct memory access) system 420, and a boot ROM (read only memory) 424. System 400 may further provide Ethernet access via an Ethernet device 426. A USB (universal serial bus) 428 is also connected to the bus 404, along with interrupts and timers 432, I/O circuitry 434, a keypad and touch screen interface 436, and a UART (universal asynchronous receiver transmitter) 440. In this regard, it will be appreciated that the exemplary raster engine 2 and video controller of the invention may be employed in a variety of systems and applications, including those not specifically illustrated and described herein.

Video Underflow Detection and Indication

Another aspect of the invention provides apparatus and methods for detecting and/or indicating overflow conditions in a raster engine. In order to provide context for the underflow detection and indication aspects of the invention, Fig. 33 illustrates an exemplary computer system 500 having a central processing unit (CPU) 562, a memory

564, and a shared system bus 566 providing an interface therebetween. A video frame buffer 568a may interface with the bus 566 via a bus interface 570. Alternatively, a frame buffer 568b may be provided in a portion of main memory 564, wherein the beginning of video lines may be located on any 32 bit word boundary. An exemplary raster engine 502 may be operatively connected with the bus 566 for receiving video data therefrom for rendering to a display device 572. In addition, the bus 566 (*e.g.*, including address and data busses) may provide access to the various control registers in raster engine 502. The bus 566 may be further shared between the CPU 562 and other devices in the system 500. For example, a direct memory access (DMA) controller 584 interacts with the bus 566, wherein the DMA controller 584 may be used as an integrated drive electronics (IDE) interface for a hard disk or CD ROM device (not shown). Other system devices may also be accessed via the system bus 566, such as an Ethernet MAC 588, a USB host controller 590, and a graphics accelerator 592.

In one form of operation, video data is transferred from the frame buffer 568b in the system memory 564 across the system bus 566, and moved to the display device 572 by the raster engine 502. The frame buffer 568b may thus be part of the same system memory 564 used by the CPU 562 for various purposes, or may be a separate video memory 568a. Where a separate frame buffer 568a is employed, the raster engine 502 may obtain video data from the frame buffer 568a via the bus interface 570 and the shared system bus 566, or alternatively via an isolated or dedicated bus 504. In this case, a frame buffer bus interface (not shown) may be provided to interface the frame buffer 568a with the isolated bus 504. When the raster engine 502 needs to access a display image store in a frame buffer 568a or 568b (*e.g.*, collectively referred to as 568) via the shared system bus 566, contention for or excessive loading of the shared bus 566 may cause the raster engine 502 to become starved for video data (*e.g.*, underflow). The configuration in which such a system bus (*e.g.*, bus 566) is shared between the CPU 562 and other system devices (*e.g.*, some of which may be "masters" in a multiple master configuration) may sometimes be referred to as a unified memory architecture (UMA) system.

Where two or more devices are in contention for access to the shared bus 566 (*e.g.*, and in particular for access to the shared system memory 564), there is a potential for the raster engine 502 to become starved for video data from the frame buffer 568, for example, under excess loading on the bus 566, or during extremely long burst operations.

When the Raster engine 502 becomes starved, undesired visual defects such as jittering, shifting, flashing, and blank-outs may occur in the video image rendered by the display device 572. According to one aspect of the present invention, the exemplary raster engine 502 may comprise an underflow detection system, which detects underflow conditions in the raster engine 502 (*e.g.*, where the raster engine 502 is or is about to become starved for video data), and provides an underflow indication 580.

The underflow indication 580 may comprise, for example, and underflow signal or interrupt, which may be provided to the system CPU 562. The CPU 562 may in turn implement remedial techniques, such as methods to balance bus loading or limit burst sizes on the shared bus 566, in order to reduce or minimize the occurrence of such underflow conditions in the raster engine 502. Although the CPU 562 may be able to detect certain bus behavior and performance measures with respect to the shared system bus 566 (*e.g.*, bus loading, etc.), the CPU 562 is not able to unilaterally determine whether an underflow condition exists or is about to occur in the raster engine 502, absent the underflow indication 580. It will be further noted that where the raster engine 502 obtains video data from the frame buffer 568a via the isolated bus 504, the CPU 562 may not even be able to detect excess bus loading, error, or lock-up conditions on the isolated bus 504. A system watchdog may not detect locked up activity on the isolated bus 504 unless the shared memory sub-system is also locked up as well. Thus, the provision of the underflow indication 580 from the raster engine 502 in accordance with the invention provides significant advantages in reducing or eliminating underflow conditions and the deleterious display effects associated with such conditions.

Referring also to Fig. 34, a portion of the exemplary raster engine 502 is illustrated having a first in first out (FIFO) memory system 516 interfacing a host bus such as the system bus 566 in the computer system 500 with the raster engine 502. The FIFO 516 includes a dual port memory 532 and is adapted to obtain video data from a frame buffer (*e.g.*, frame buffer 568) via the host bus 566 (*e.g.*, or via a dedicated or isolated bus such as bus 504) and to provide video data to a video pipeline (*e.g.*, as illustrated and described above with respect to Fig. 1). Raster engine 502 further includes an input address counter 534 having an input counter value (not shown) indicative of video data obtained from the frame buffer via bus 566 and an output counter 536 having an output counter value (not shown) indicative of video data provided to the video pipeline. The input and output of the FIFO memory system 516 may be operated

according to separate clocks. For instance, the input of video data from the host bus 566, including the operation of the input counter 534, may be performed according to a host clock 600, and the output of video data to the video pipeline together with the operation of the output counter 536 may be performed according to a video clock 602.

A control logic system 538 is associated with the FIFO memory 516 and is adapted to provide an underflow indication 580, such as an interrupt to the CPU 562, according to the values of the input and output counters 534 and 536, respectively. In addition, the control logic system 538 provides interfacing to a video image line output scanner and transfer interface (VILOSATI) 514, which may connect to a dedicated DMA port on an SDRAM controller (not shown). The VILOSATI 514 reads video image data from memory (*e.g.*, frame buffer 568) and transfers the image data to the video FIFO 516. VILOSATI 514 keeps track of image location, width, and depth for both progressive and dual scanned images, and responds to controls (*e.g.*, FULL, DS_FULL) from the FIFO 516 for more video data. For example, during a single scan operation, when the FIFO 516 has room for a 16 word burst, the FULL signal may be inactive and the VILOSATI 514 attempts to initiate a burst. The VILOSATI 514 will initiate appropriate size transfers and bursts in order to get to a 16 word boundary. After this point, VILOSATI 514 will perform transfers more efficiently using 16 word long bursts. When the FIFO 516 is full (*e.g.*, 40 to 64, 32 bit words), the current burst is completed, and no further data is requested. When the FIFO 516 signals that it has room for a burst again, the image reading process from the frame buffer continues.

For dual scan operation, the FIFO 516 may be split in two and operates with a separate FULL indicator for each half. In this mode, the FULL signal and a DS_FULL indicator (not shown) trigger from 12 to 32 words. The VILOSATI 514 continues to service the video FIFO 516 until it has transferred an entire screen image (*e.g.*, a frame) from memory. The video FIFO 516 is used to buffer video data transferred from the frame buffer memory (*e.g.*, of frame buffer 668) to the video output system without stalling the video data stream of the raster engine 502. During dual scan mode, wherein the display requires scan out of the bottom and top half of the screen at the same time, top half (or bottom half) information is stored in every other location in the FIFO 516. In progressive scan mode wherein video data is scanned out as a single progressive image, the FIFO data is stored sequentially.

The FIFO output data bus 533 is 64 bits wide and can output even and odd words

on both the upper and lower half of the bus. Writes to the FIFO 516 advance the input index counter 534, while reads from the FIFO 516 advance the output index counter 536. The input and output counters 534 and 536 are compared to generate the FULL and DS_FULL output controls to the VILOSATI 514, as well as to determine whether an underflow condition exists or is likely to occur. The N_CLR signal resets both the input and output input and output counters 534 and 536 to 0, for example, at the end of a video frame. The control logic system 538 in the FIFO system 516 includes an underflow detection and indication system, which operates to detect an underflow of the FIFO 516 (*e.g.*, dual port RAM 532) and/or a near underflow condition therein, and to provide the Underflow_INT signal 580 according to the detected underflow condition.

The underflow system of the FIFO control logic 538 may include, for example, comparison logic for comparing the values of input and output counters 534 and 536, respectively, and for making a determination of whether an underflow condition exists or is anticipated. The Underflow_INT indication 580 may be advantageously provided to a host processor (*e.g.*, CPU 562) whereby steps to balance bus loading or to limit burst sizes may be taken. This feature is particularly advantageous where the raster engine interface with the frame buffer memory is via a bus isolated from that of the host processor. In this situation, the host CPU (*e.g.*, CPU 562) may not be able to independently detect or sense bus loading conditions resulting in a starving raster engine 502. Thus, the invention provides for early indication to the host processor 562, whereby elimination or reduction of raster engine underflow conditions may be achieved. The underflow indication 580 may further be used to indicate a lockup condition in the raster engine 502.

Referring also to Fig. 35, further details of the exemplary control logic system 538 are illustrated. The control logic system 538 comprises underflow detection and indication components to provide the underflow indication 580 when an underflow condition exists and also when an underflow condition is anticipated. For example, a threshold value register 604 may be provided, which may be programmable by a host processor (*e.g.*, CPU 562), wherein the control logic system 538 obtains a threshold value (*e.g.*, 3) from the threshold value register 604, and compares the threshold value with the difference between the input and output counter values. To accomplish this, the exemplary system 538 comprises a subtractor 606 receiving an input counter value (*e.g.*, 6 bits) from the input counter 534 and an output counter value from the output counter

536. The subtractor 606 provides a difference value 608 (*e.g.*, input counter value - output counter value) to a comparator 610.

The comparator 610 performs a comparison of the difference value 608 with the threshold value from the threshold value register 604. The underflow indication 580 may be selectively provided (*e.g.*, to host CPU 562) to indicate an existing underflow condition when the input and output counter values are equal, and to indicate an anticipated underflow condition when the input and output counter values are within the threshold value of each other. Additional components may be provided in the system 538 to ensure that the underflow indication is accurate in situations where the FIFO 516 obtains video data from the frame buffer (*e.g.*, 568) according to the host clock 600 and provides video data to the video pipeline according to the video clock 602, as described further hereinafter. This may be accomplished, for example, by providing an underflow condition 580 when the first input and output counter values are equal or within the threshold value of each other for at least two cycles of the host clock 600.

The comparator 610 may provide an input signal to a first voting flip-flop 620 operating according to the host clock 600, as well as to an AND gate 622 providing a logical ANDing of the output of the flip-flop 620 and the signal from the comparator 610. The result of this AND operation is provided to a second voting flip-flop 624. The outputs of the first and second flip-flops 620 and 624 are provided as inputs to an AND gate 626 which provides and input to a third flip-flop 628. The third flip-flop 628 accordingly provides a signal when the values of the input and output counters 534 and 536, respectively, are within the threshold value of each other for at least two cycles of the host clock 600.

The third flip-flop 628 provides the underflow indication 580 via a tri-state buffer 630, whereby the provision of the indication 580 may be selectively enabled or disabled according to an enable signal 632. Thus, for example, the underflow indication may be selectively suppressed in situations where the values of the counters 534 and 536 are within the threshold value of each other for reasons other than actual (or anticipated) underflow. One such situation is when the FIFO 516 is emptied at the end of a frame. Another is during startup of the FIFO 516, before any data has been obtained from the host bus 566. In this regard, the enable signal 632 may be provided by an RS flip-flop 640 which is set according to a FULL signal 642 from a FIFO flags component 643, and reset according to a Vertical End of Frame signal 644, wherein the signals 642 and 644

are derived from the input and output counters 534 and 536, respectively. For instance, the control logic system 538 may perform a subtraction of input and output counters 534 and 536, and the result may be compared with a threshold (not shown) to generate the signal 644. In this manner, the underflow indication may be suppressed until the FIFO 516 fills once after being initially empty, and thereafter once a frame boundary has been achieved. It will be appreciated that the enable signal 632 may be generated according to other appropriate signals and system conditions, in order to reduce or prevent false underflow indications, in accordance with the invention.

According to another aspect of the invention, the system 538 may further provide selective underflow indication when the FIFO 516 is operating in dual scan mode. Referring now to Fig. 36, another implementation of the exemplary control logic system 538 obtains counter values from first input and output counters 534a and 536a, respectively (*e.g.*, counting for the upper display portion in dual scan mode), as well as from second input and output counters 534b and 536b, respectively (*e.g.*, counting for the lower display portion in dual scan mode). It will be appreciated that the first and second input counters (*e.g.*, counters 534a and 534b) may be separate counters, or be included within a single counter device, such as where a first set of counter output bits represents the first input counter 534a, and a second set of bits represents the second input counter 534b. Similarly, the first and second output counters 536a and 536b may be separate devices or portions of a single device.

A first subtractor 608a receives input and output counter values from first input and output counters 534a and 536a, respectively, and provides a difference value 608a to a comparator 610a. The comparator 610a compares the difference value 608a with a threshold value from a threshold register 604a to determine whether the values of the counters 534a and 536a are within the threshold value of each other. Logic components 620a, 622a, 624a, 626a, and 628a operate in similar fashion to the components 620, 622, 624, 626, and 628, respectively, of Fig. 35, in order to provide a first signal 650a according to whether the values of the counters 534a and 536a are within the threshold value of each other for at least two cycles of the host clock 600.

With respect to second input and output counters 534b and 536b, a second subtractor 608b receives input and output counter values from second input and output counters 534b and 536b, respectively, and provides a difference value 608b to a comparator 610b. The comparator 610b compares the difference value 608b with a

threshold value from a threshold register 604b to determine whether the values of the counters 534b and 536b are within the threshold value of each other. It will be appreciated that the comparators 610a and 610b may alternatively obtain threshold values from a single threshold register, whereby the determinations of an anticipated underflow in the upper and lower dual scan memory portions of the FIFO 516 may be made according to the single threshold value. Logic components 620b, 622b, 624b, 626b, and 628b operate in similar fashion to the components 620, 622, 624, 626, and 628, respectively, of Fig. 35, in order to provide a second signal 650b according to whether the values of the counters 534b and 536b are within the threshold value of each other for at least two cycles of the host clock 600.

The first and second signals 650a and 650b, respectively, are provided as inputs to an OR gate 660, whose output is provided to the tri-state buffer 630, whereby the provision of the indication 580 may be selectively enabled or disabled according to the enable signal 632. As described above with respect to Fig. 35, the underflow indication may be selectively suppressed in situations where the values of the counters 534a and 536a (*e.g.*, or the values of counters 534b and 536b) are within the threshold value of each other for reasons other than actual (or anticipated) underflow (*e.g.*, such as where the FIFO 516 is emptied at the end of a frame or during startup of the FIFO 516, before any data has been obtained from the host bus 566).

Thus, the underflow indication 580 indicates an existing or anticipated underflow condition in the FIFO 516 when the first (*e.g.*, upper) input and output counter values are within the threshold value of each other or when the second (*e.g.*, lower) input and output counter values are within the threshold value of each other. The indication may be further refined by indicating an overflow when such conditions have been met for a number of consecutive cycles of the host clock 600 (*e.g.*, two cycles). It will be appreciated that the underflow indication 580 may further indicate a lockup condition in the raster engine FIFO 516 in accordance with the invention.

The various aspects of the invention may be achieved by various forms and combinations of control logic components, all, some, or none of which may be programmable. For example, the threshold value may, but need not be programmable (*e.g.*, via the threshold register 604). Also, the underflow indication may be based on two cycles of the host clock 600, or any number of such cycles (*e.g.*, including zero). Furthermore, one or more of the values generated in the underflow determination may be

stored in registers, for example, such as the difference value 608 provided by the subtractor 606. All such variations in form of the circuitry, programmability, and/or logic for implementing the aspects of the invention are deemed as falling within the scope of the invention, including the appended claims.

The underflow detection system of the invention (*e.g.*, including the exemplary control logic system 538) thus provides an underflow indication, which may be provided to a system processor (*e.g.*, CPU 562) for informational use, and/or for use in performing one or more remedial actions (*e.g.*, to reduce bus loading, etc.). Such a host processor may selectively perform one or more such remedial actions based on various factors, including the nature of the underflow indication or indications. For example, where the underflow indication (*e.g.*, indication 580) is provided as a processor interrupt, the host processor may take appropriate action according to the frequency of such underflow interrupts. In this regard, infrequent or spurious interrupts may mean that there is a long burst transaction, which is starving the raster engine (*e.g.*, raster engine 502), which may be more likely in applications that demand high video bandwidth. On the other hand, frequent interrupts may indicate or be interpreted by the processor as meaning that the system bus is just out of bandwidth for the video mode that the raster engine is trying to support. Moreover, continuous interrupts may indicate that the video mode is unsupportable or that raster transfer on an isolated bus is locked up, and that the video and/or memory subsystem needs to be reset to a know state. If the interrupt occurs spuriously, depending on frequency, this may indicate that either the system is running out of bus bandwidth, that a master is holding the bus for too long, or both.

The underflow detection system, moreover, is adapted to interact between two clock domains (*e.g.*, the host clock 600 and the video clock 602), as well as providing support for dual scan display operating modes. In addition, an underflow condition indication may be selectively suppressed when the FIFO is supposed to be empty. The invention provides for selective underflow indication when the FIFO has already been in use and is almost empty. This helps to indicate being on the verge of inadequate bandwidth. In normal operation, once the FIFO has started filling, the output counter (*e.g.*, counter 536) should always stay behind the input counter (*e.g.*, counter 534) until the end of the screen. At this point, they could be the same. At the end of the frame, both counters are cleared for the beginning of the next frame. Where the FIFO locations are a base 2 multiple, the rollover and detection logic may be implemented by a

subtractor (*e.g.*, subtractor 606) without any math adjustments.

Another aspect of the present invention provides a methodology for detecting and indicating an underflow condition in a raster engine. Referring now to Fig. 37, an exemplary method 700 for detecting underflow conditions in a raster engine is illustrated and described hereinafter. Although the method 700 is illustrated and described herein as a series of steps, it will be appreciated that the present invention is not limited by the illustrated ordering of steps, as some steps may occur in different orders and/or concurrently with other steps apart from that shown and described herein, in accordance with the invention. In addition, not all illustrated steps may be required to implement a methodology in accordance with the present invention. Moreover, it will be appreciated that the method 700 may be implemented in association with the apparatus and systems illustrated and described herein as well as in association with other systems not illustrated, whether hardware, software, or combinations thereof.

Beginning at step 702, a determination is made at step 706 as to whether the FIFO is empty (*e.g.*, such as at startup, or at the end of a frame). If not, the method 700 proceeds to step 712 as described hereinafter. If the FIFO is empty at step 706, the raster engine waits at step 708 until the FIFO is full at step 710. It will be appreciated that no underflow indication is provided at steps 706-710 to avoid false underflow indications when the FIFO is emptied, such as during power up or at the end of a frame.

At step 712, input and output counter values are obtained (*e.g.*, from input and output counters 534 and 536, respectively) and the difference therebetween is determined at step 714 (*e.g.*, using a subtractor). A threshold value is then obtained at step 716 (*e.g.*, from a programmable register), and at step 718 the difference value computed at step 714 is compared with the threshold value obtained at step 716. A determination is then made at step 720 as to whether the difference value is less than or equal to the threshold value. If not, the method 700 returns to step 706 described above. However, if the difference value is less than or equal to the threshold value (*e.g.*, the input and output counter values are within the threshold value of each other), the method proceeds to step 722, whereat an underflow indication is provided. The indication may include, for example, providing an interrupt signal to a host processor (*e.g.*, CPU 562). The method 700 then returns to 706 as described above.

Although the invention has been shown and described with respect to certain implementations, it will be appreciated that equivalent alterations and modifications will

occur to others skilled in the art upon the reading and understanding of this specification and the annexed drawings. In particular regard to the various functions performed by the above described components (assemblies, devices, circuits, systems, etc.), the terms (including a reference to a "means") used to describe such components are intended to correspond, unless otherwise indicated, to any component which performs the specified function of the described component (*i.e.*, that is functionally equivalent), even though not structurally equivalent to the disclosed structure, which performs the function in the herein illustrated exemplary applications and implementations of the invention.

In addition, while a particular feature of the invention may have been disclosed with respect to only one of several aspects or implementations of the invention, such a feature may be combined with one or more other features of the other implementations as may be desired and advantageous for any given or particular application. Furthermore, to the extent that the terms "includes", "including", "has", "having", and variants thereof are used in either the detailed description or the claims, these terms are intended to be inclusive in a manner similar to the term "comprising" and its variants.